pong: fast analysis and visualization of latent clusters in population genetic data

Aaron A. Behr [1,2,*], Katherine Z. Liu[2,†], Gracie Liu-Fang[3,†], Priyanka Nakka[1,4], and Sohini Ramachandran[1,4*]

November 14, 2015

[1]Department of Ecology and Evolutionary Biology, Brown University, Providence, RI, USA
[2]Department of Computer Science, Brown University, Providence, RI, USA
[3]Computer Science Department, Wellesley College, Wellesley, MA, USA
[4]Center for Computational Molecular Biology, Brown University, Providence, RI, USA

Abstract

Motivation: A series of methods in population genetics use multilocus genotype data to assign individuals membership in latent clusters. These methods belong to a broad class of mixed-membership models, such as latent Dirichlet allocation used to analyze text corpora. Inference from mixed-membership models can produce different output matrices when repeatedly applied to the same inputs, and the number of latent clusters is a parameter that is varied in the analysis pipeline. For these reasons, quantifying, visualizing, and annotating the output from mixed-membership models are bottlenecks for investigators.

Results: Here, we introduce pong, a network-graphical approach for analyzing and visualizing membership in latent clusters with a D3.js interactive visualization. We apply this new method to 225,705 unlinked genome-wide single-nucleotide variants from 2,426 unrelated individuals in the 1000 Genomes Project, and show that pong outpaces current solutions by more than an order of magnitude in runtime while providing a customizable and interactive visualization of population structure that is more accurate than those produced by current tools.

Availability: pong is freely available and can be installed using the Python package management system `pip`.

Contact: aaron_behr@alumni.brown.edu, sramachandran@brown.edu

1 Introduction

A series of generative models known as mixed-membership models have been developed that model grouped data, where each group is characterized by a mixture of latent components. One well-known example of a mixed-membership model is latent Dirichlet allocation (Blei et al., 2003), in which documents are modeled as a mixture of latent topics. Another widely used example is the model implemented in the population-genetic program Structure (Pritchard et al., 2000; Falush et al., 2003; Hubisz et al., 2009; Raj et al., 2014), where individuals are assigned to a mixture of latent *clusters*, or populations, based on multilocus genotype data.

In this paper, we focus on the population-genetic application of mixed-membership models, and refer to this application as *clustering inference*; see Novembre (2014) for a review of multiple population-genetic clustering inference methods, including Structure. In Structure's Bayesian Markov chain Monte Carlo (MCMC) algorithm, individuals are modeled as deriving ancestry from K clusters, where

*To whom correspondence should be addressed.
†These authors contributed equally to this work.

the value of K is user-specified. Each cluster is constrained to be in Hardy-Weinberg equilibrium, and clusters vary in their characteristic allele frequencies at each locus. Clustering inference using genetic data is a crucial step in many ecological and evolutionary studies. For example, identifying genetic subpopulations provides key insight into a sample's ecology and evolution (Bryc et al., 2010; Glover et al., 2012; Moore et al., 2014), reveals ethnic variation in disease phenotypes (Moreno-Estrada et al., 2014), and reduces spurious correlations in genome-wide association studies (Price et al., 2006; Patterson et al., 2006; Galanter et al., 2012).

For a given multilocus genotype dataset with N individuals and K clusters, the output of a single algorithmic run of clustering inference is an $N \times K$ matrix, denoted as Q, of *membership coefficients*; these coefficients can be learned using a supervised or unsupervised approach. Membership coefficient q_{ij} is the inferred proportion of individual i's alleles inherited from cluster j. The row vector $\vec{q}_{i\,\bullet}$ is interpreted as the genome-wide ancestry of individual i, and the K elements of $\vec{q}_{i\,\bullet}$ sum to 1. Each column vector $\vec{q}_{\bullet\,j}$ represents membership in the jth cluster across individuals.

Although covariates — such as population labels, geographic origin, language spoken, or method of subsistence — are not used to infer membership coefficients, these covariates are essential for interpreting Q matrices. Given that over 14,000 studies have cited STRUCTURE to date, and 100 or more Q matrices are routinely produced in a single study, investigators need efficient algorithms that enable accurate processing and interpretation of output from clustering inference.

Algorithms designed to process Q matrices face three challenges. First, a given run, which yields a single Q matrix, is equally likely to reach any of $K!$ column-permutations of the same collection of estimated membership coefficients due to the stochastic nature of clustering inference. This is known as *label switching*: for a fixed value of K and identical genetic input, column $\vec{q}_{\bullet\,j}$ in the Q matrix produced by one run may not correspond to column $\vec{q}_{\bullet\,j}$ in the Q matrix produced by another run (Stephens, 2000; Jasra et al., 2005; Jakobsson and Rosenberg, 2007).

Second, even after adjusting for label switching, Q matrices with the same input genotype data and the same value of K may differ non-trivially. This is known as *multimodality* (Jakobsson and Rosenberg, 2007), and occurs when multiple sets of membership coefficients can be inferred from the data. We refer to runs that, despite identical inputs, differ non-trivially as belonging to different *modes*. For a fixed value of K, a set of runs grouped into the same mode based on some measure of similarity can be represented by a single Q matrix in that mode. The complete characterization of modes present in clustering inference output gives unique insight into genetic differentiation within a sample.

A third complication arises for interpreting clustering inference output when the input parameter K is varied (all other inputs being equal): there is no column-permutation of ~~an~~ $Q_{N\times K}$ matrix that exactly corresponds to any $Q_{N\times(K+1)}$ matrix. We refer to this as the *alignment-across-K* problem. A common pipeline when applying clustering inference methods to genotype data is to increment K from 2 to some user-defined maximum value K_{max}, although some clustering inference methods also assist with choosing the value of K that best explains the data (Huelsenbeck et al., 2011; Raj et al., 2014). K_{max} can vary a great deal across studies (e.g., $K_{max} = 5$ in Glover et al. (2012); $K_{max} = 20$ in Moreno-Estrada et al. (2014)). Accurate and automated analysis of clustering inference output across values of K is essential for both understanding a sample's evolutionary history and model selection.

The label-switching, multimodality, and alignment-across-K challenges must all be resolved in order to fully and accurately characterize genetic differentiation and shared ancestry in a dataset of interest. Here, we present pong, a new algorithm for fast post-hoc analysis of clustering inference output from population genetic data combined with an interactive JavaScript data visualization using Data-Driven Documents (D3.js; https://github.com/mbostock/d3). Our package accounts for label switching, characterizes modes, and aligns Q matrices across values of K by constructing weighted bipartite graphs for each pair of Q matrices depicting similarity in membership coefficients between clusters. pong displays an accurate representative Q matrix for each mode for each value of K, clarifies differences among modes that are difficult to identify through visual inspection, and presents results that may be overlooked when interpreting output by hand. We compare pong against current solutions (*CLUMPP* by Jakobsson and Rosenberg (2007); augmented as CLUMPAK by Kopelman et al. (2015)),

and find our approach reduces runtime by more than an order of magnitude. pong has the potential to be applied broadly to identify modes, align output, and visualize output from inference based on mixed-membership models.

2 Algorithm

2.1 Overview

Figure 1 displays a screenshot of pong's visualization of population structure in the 1000 Genomes data (phase 3, Consortium (2015); final variant set released on November 6, 2014) based a set of 20 runs ($K = 4, 5$) from clustering inference with ADMIXTURE (Alexander et al., 2009). In order to generate visualizations highlighting similarities and differences among Q matrices, pong generates weighted bipartite graphs connecting clusters between runs within and across values of K (Section 2.3). Our goal of matching clusters across runs is analogous to solving the combinatorial optimization problem known as the Assignment Problem (Manber, 1989), for which numerous algorithms exist (Kuhn, 1955, 1956; Munkres, 1957). pong's approach of comparing clusters dramatically reduces runtime relative to existing methods.

Consider two Q matrices, $\mathcal{Q} = [q_{ij}]$ and $\mathcal{R} = [r_{ij}]$. Each weighted bipartite graph $G(\mathcal{Q}, \mathcal{R}) = (\{\vec{q}_{\bullet j}\} \cup \{\vec{r}_{\bullet k}\}, E)$ encodes pairwise similarities between clusters (column vectors) in $\mathcal{Q}$ and clusters in $\mathcal{R}$. Edges in G are weighted according to a similarity metric computed between clusters (detailed in Section 2.2). We define an *alignment* of $\mathcal{Q}$ and $\mathcal{R}$ as a bipartite perfect matching of their column vectors. pong's first objective is to find the maximum-weight alignment for each pair of runs for a fixed value of K (Section 2.3). This information is used to identify modes within K, and we randomly choose a representative run (Q matrix) for each mode found in clustering inference. We call the mode containing the most runs within each value of K the *major mode* for that K value (Figure 1A; ties are decided uniformly at random). pong's second objective is to find the maximum-weight alignment between the representative run of each major mode across values of K (Section 2.4; Figures 1B, S1). Note that identifying the maximum-weight alignment within and across K inherently solves the label switching problem. Lastly, pong colors the visualization and highlights differences among modes based on these maximum-weight alignments.

2.2 Cluster similarity metrics

We implemented and tested several metrics for cluster similarity. The default metric used by pong, $\mathcal{J}$ (Equation 1), is derived from the Jaccard index used in set comparison. For a given pair of clusters $\{\vec{q}_{\bullet a}, \vec{r}_{\bullet b}\}$, let N^* be the set of indices for which at least one of $\{\vec{q}_{\bullet a}, \vec{r}_{\bullet b}\}$ has a nonzero entry; that is, $N^* = \{i \in \{1, \ldots, N\} : q_{ia} + r_{ib} > 0\}$. Then,

$$\mathcal{J}(\vec{q}_{\bullet a}, \vec{r}_{\bullet b}) = 1 - \sqrt{\frac{\sum_{i \in N^*} (q_{ia} - r_{ib})^2}{2|N^*|}} \tag{1}$$

$\mathcal{J}$ is designed to emphasize overlap in membership coefficients while ignoring overlap in nonmembership (i.e., individuals with membership coefficients of 0 in the clusters under comparison). Although we recommend using $\mathcal{J}$, pong implements other similarity metrics: G' (as used in *CLUMPP* Jakobsson and Rosenberg (2007)), the average sum of squared differences between $\vec{q}_{\bullet a}$ and $\vec{r}_{\bullet b}$ (subtracted from 1), and average Manhattan distance (subtracted from 1). pong's implementation is designed such that users familiar with Python and NumPy can add their own similarity metrics to the source code if desired.

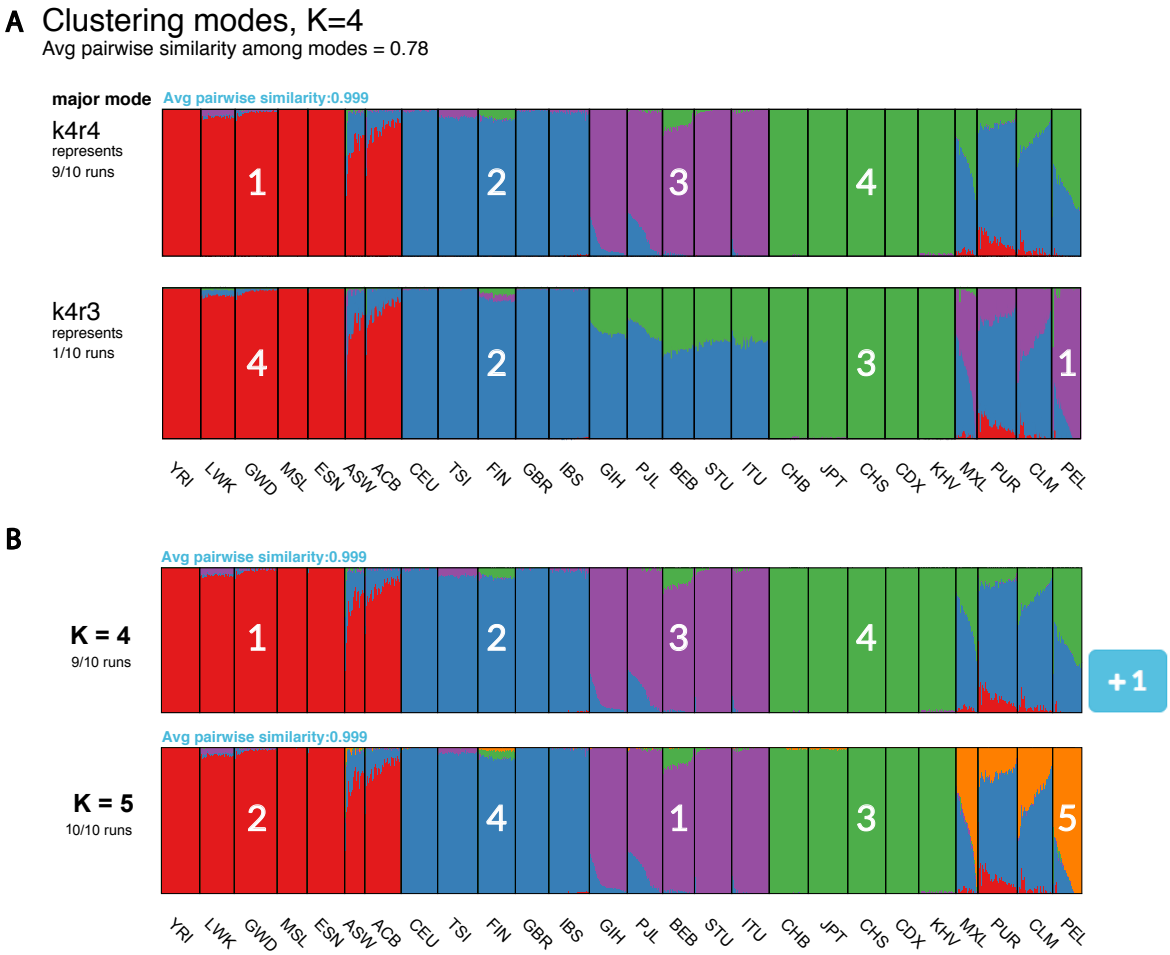


Figure 1: pong's front end produces a D3.js visualization of maximum-weight alignments between runs, shown here for 20 Q matrices produced from clustering inference with ADMIXTURE (Alexander et al., 2009) applied to 1000 Genomes data (phase 3; Consortium (2015)). Each individual's genome-wide ancestry within a barplot is depicted by K stacked colored lines. The left-to-right order of individuals is the same in each barplot. The barplots here are annotated with numbers indicating which column of the underlying Q matrix is represented by a given cluster. **A:** Characterizing modes at $K = 4$ by displaying the representative run of the major mode (here, k4r4) and the representative run of each minor mode. Population codes are shown at the bottom. **B:** The maximum-weight alignment for the representative run for the major mode at $K = 4$ (k4r4, panel A) to that at $K = 5$. Membership in cluster 4 at $K = 4$ represents shared ancestry in East Asian and admixed American populations, and has been partitioned into Clusters 3 and 5 (representing East Asian and Native American ancestry, respectively) in the representative run of the major mode at $K = 5$.

2.3 Aligning runs for a fixed value of K and characterizing modes

In order to identify modes in clustering inference for a fixed value of $K = k$, pong first uses the Munkres algorithm (Munkres, 1957) to find the maximum-weight alignment between each pair of runs at $K = k$ (Figure 2A). Next, for each value k, pong constructs another graph $G_k = (\{Q_{N\times k}\}, E)$, where each edge connects a pair of runs, and the weight of a given edge is the average edge weight in the maximum-weight alignment for the pair of runs that edge connects. (The edge weight between a run and itself is 1.) Each pair of runs has a maximum-weight alignment, and we define *pairwise similarity* for a pair of runs as the average edge weight in the maximum-weight alignment; the edge weight for a pair of runs in G_k encodes their pairwise similarity. We use the average edge weight to compute pairwise similarity instead of the sum of edge weights so that edges are comparable across values of K. If a pair of runs has pairwise similarity less than 0.97 (by default; this threshold can be varied), the edge connecting that pair of runs is not added to G_k; this imposes a lower bound on the pairwise similarity between two runs in the same mode. pong defines modes as disjoint cliques in G_k, thereby solving the multimodality problem. Once cliques are identified, a run is chosen at random to be the representative run for each mode at $K = k$, which enables consistent visualization of clustering inference output within each value of K.

2.4 Aligning a $Q_{N\times K}$ matrix to a $Q_{N\times(K+1)}$ matrix

Consider two Q matrices $\mathcal{T}_{N\times k}$ and $\mathcal{U}_{N\times(k+1)}$ where $\mathcal{T}$ and $\mathcal{U}$ represent the major modes at $K = k$ and $K = (k + 1)$, respectively. No perfect matching can be found between the clusters in $\mathcal{T}$ and the clusters in $\mathcal{U}$ because these matrices have different dimensions. In order to align these matrices, pong leverages the fact that column vectors of membership coefficients are partitioned as K increases and summed as K decreases (Figure 1B).

For the pair of clusters $\vec{u}_{\bullet a}$ and $\vec{u}_{\bullet b}$ in $\mathcal{U}$, we define the *union node* $\vec{u}_{\bullet\{a,b\}} = \sum_{i=1}^{N} u_{ia} + u_{ib}$. pong then constructs the matrix $\mathcal{U}(a \cup b)$, which contains the clusters $\vec{u}_{\bullet i}$ for $i \neq a, b$ and the union node $\vec{u}_{\bullet\{a,b\}}$. Therefore, the dimension of $\mathcal{U}(a \cup b)$ is $N \times K$, which is the same as the dimension of $\mathcal{T}$ (Figure 2B). pong then finds the maximum-weight alignment between $\mathcal{T}$ and $\mathcal{U}(a\cup b)$ using the Munkres algorithm (Munkres, 1957). After finding the maximum-weight alignment for each pair of matrices $\mathcal{T}$ and $\mathcal{U}(i \cup j)(i \neq j)$, the alignment that has the greatest average edge weight across all these $\binom{k+1}{2}$ alignments is then used to solve the alignment-across-K problem. pong begins alignment across K between the representative runs of the major modes at $K = 2$ and $K = 3$ and proceeds through aligning $K = K_{max} - 1$ and $K = K_{max}$.

3 Implementation

pong's back end is written in Python. While providing covariates is strongly advised so visualizations can be annotated with relevant metadata, pong only requires one tab-delimited file containing: (*i*) a user-provided identification code for each run (e.g., k4r4 in Figure 1A), (*ii*) the K-value for each run, and (*iii*) the relative path to each Q matrix. pong is executed with a one-line command in the terminal, which can contain a series of flags to customize certain algorithmic and visualization parameters.

After characterizing modes and aligning runs, pong starts a Tornado web server (http://www.tornadoweb.org/) instance to host the visualization. The user is prompted to open a web browser and navigate to a specified port, and the user's actions in the browser window lead to the exchange of data, such as Q matrices, via web sockets. These data are bound to and used to render the visualization.

pong's front-end visualization is implemented in D3.js. pong's main visualization displays the representative Q matrix for the major mode for each value of K as a Scalable Vector Graphic (SVG), where each individual's genome-wide ancestry is depicted by K stacked colored lines. Each SVG is annotated with its value of K, the number of runs grouped into the major mode, and the average pairwise similarity across all pairs of runs in the major mode (Figure 1B).

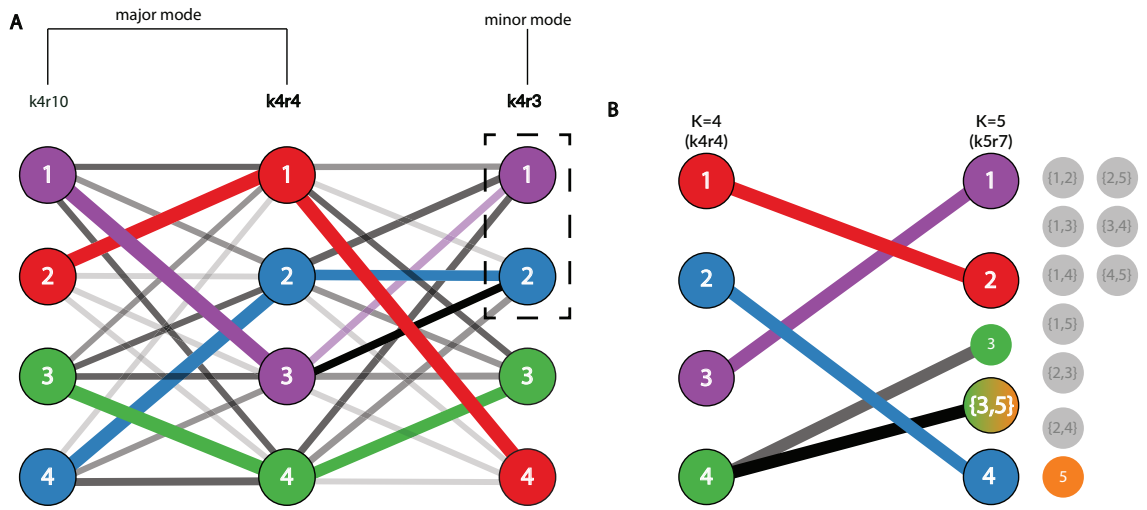


Figure 2: pong's back-end model for the alignment of Q matrices, shown here from clustering inference with ADMIXTURE (Alexander et al., 2009) applied to 1000 Genomes data (phase 3; Consortium (2015)). Panel labels correspond to panels in Figure 1, and numbers in graph vertices correspond to the clusters labeled in Figure 1. **A:** Characterizing modes from three runs of clustering inference at $K = 4$, the smallest K value with multiple modes for this dataset. Edge thickness corresponds to the value of the similarity metric $\mathcal{J}$ (Equation 1), while edge opacity ranks connections for a cluster in run k4r4 to a cluster in run k4r3 (or in run k4r10). Note that both cluster 2 and 3 in k4r4 are most similar based on metric $\mathcal{J}$ to cluster 2 in k4r3; in order to find the maximum-weight perfect matching between the runs, pong matches cluster 3 in k4r4 with cluster 1 in k4r3. Bold labels indicate representative runs for the two modes. Seven other runs (not displayed for ease of visualization) are grouped in the same mode as k4r4 and k4r10; these nine runs comprise the major mode at $K = 4$ (Figure 1A). k4r3 is the only run in the minor mode (Figure 1A). **B:** Alignment of representative runs for the major modes at $K = 4$ to $K = 5$. $\binom{5}{2} = 10$ alignments are constructed between k4r4 and k5r7 (the representative run of the major mode at $K = 5$), constrained by the use of exactly one union node at $K = 5$. Of these 10 alignments, the alignment with maximum edge weight matches cluster 4 in k4r4 to the sum of clusters 3 and 5 in k5r7. The second-best match for cluster 4 in k4r4, using $\mathcal{J}$ (Equation 1), is cluster 3 in k5r7. The best matching for all other clusters are shown and informs the coloring of pong's visualization (see Figure 1B).

For each value of K, a button is displayed to the right of the main visualization indicating the number of minor modes, if any exist (Figure 1B). Clicking on the button opens a pop-up dialog box consisting of barplots for the representative run of each mode within the K value, and each plot is annotated with the representative run's user-provided identification code and the number of runs in each mode (Figure 1A). A dialog header reports the average pairwise similarity among pairs of representative runs for each mode, if there is more than one mode. Users can print or download any barplot in pong's visualization from the browser window.

What truly sets pong apart from existing methods for the graphical display of population structure is a series of interactive features, which we now detail. In the browser's main visualization, the user may click on any population — or set of populations, by holding SHIFT – to highlight the selected group's genome-wide ancestry across values of K. When mousing over a population, the population's average membership (as a percentage) in each cluster is displayed in a tooltip. Within each dialog box characterizing modes, selecting a checkbox on the top right allows the user to highlight differences between the major mode's representative plot and each minor mode's representative plot (Figure 3A). Clusters that do not differ beyond a threshold between a given major and minor mode are then shown as white in the minor mode, while the remaining clusters are shown at full opacity (Figure 3A; see also edge weights in Figure 2A).

4 Results

We ran ADMIXTURE (Alexander et al., 2009) on 225,705 genome-wide single-nucleotide variants from 2,426 unrelated individuals in the 1000 Genomes Project (phase 3, Consortium (2015); see Supplementary Information) to characterize population structure among globally distributed human populations. ADMIXTURE was run with K ranging from 2 to 10 and 10 runs per value of K. Thus, a total of 90 Q matrices were produced; Figures 1 and 2 depict pong's analysis of 20 of these runs. We applied both pong and Clumpak (Kopelman et al., 2015), the state-of-the-art method for automated post-processing and visualization of clustering inference output, to these 90 runs (partial results shown in Figure 3; see also Supplementary Figures S1,S2).

Under its default settings, pong parsed input, characterized modes and aligned Q matrices within each value of K, and aligned Q matrices across K in 17.5 seconds on a Mid-2012 MacBook Pro with 8 GB RAM. After opening a web browser, pong's interactive visualization loaded in an additional 3.2 seconds (Supplementary Figure S1 shows the main visualization).

Clumpak automatically runs *CLUMPP* (Jakobsson and Rosenberg, 2007) for each value of K as part of its pipeline, and produces visualizations within and across values of K using Distruct (Rosenberg, 2004), displaying one barplot per mode. Figure 3B shows Clumpak's reported major mode in the 1000 Genomes dataset at $K = 10$, which averages over six runs; all major modes reported by Clumpak can be viewed in Supplementary Figure S1. Using Clumpak with its default settings for post-processing of these 90 runs took 58 minutes and 18 seconds, running on Clumpak's web server (http://clumpak.tau.ac.il/).

In Figure 3A, pong identifies four modes at $K = 10$ in the 1000 Genomes dataset (phase 3). Light blue represents the cluster of membership coefficients first identified at $K = 10$ (see also Supplementary Figures S1, S2). In run k10r4 (representing 4 out of 10 runs), light blue represents British/Central European ancestry in the major mode (CEU and GBR). However, light blue represents South Asian ancestry (GIH) in 3 out of 10 runs (e.g., run k10r7), Puerto Rican ancestry (PUR) in 2 out of 10 runs (e.g., run k10r3), and Han Chinese ancestry in run k10r9. pong's display of representative runs for each mode allows the user to observe and interpret multiple sets of membership coefficients inferred from the data at a given value of K.

In contrast, the minor mode Clumpak outputs (Figure 3C) is the same as pong's major mode (Figure 3A), while Clumpak's major mode reported at $K = 10$ (Figure 3B) averages over all minor modes identified by pong. The light blue in Clumpak's reported major mode could be easily misinterpreted as shared ancestry among South Asian, East Asian, and Puerto Rican individuals, when in actuality

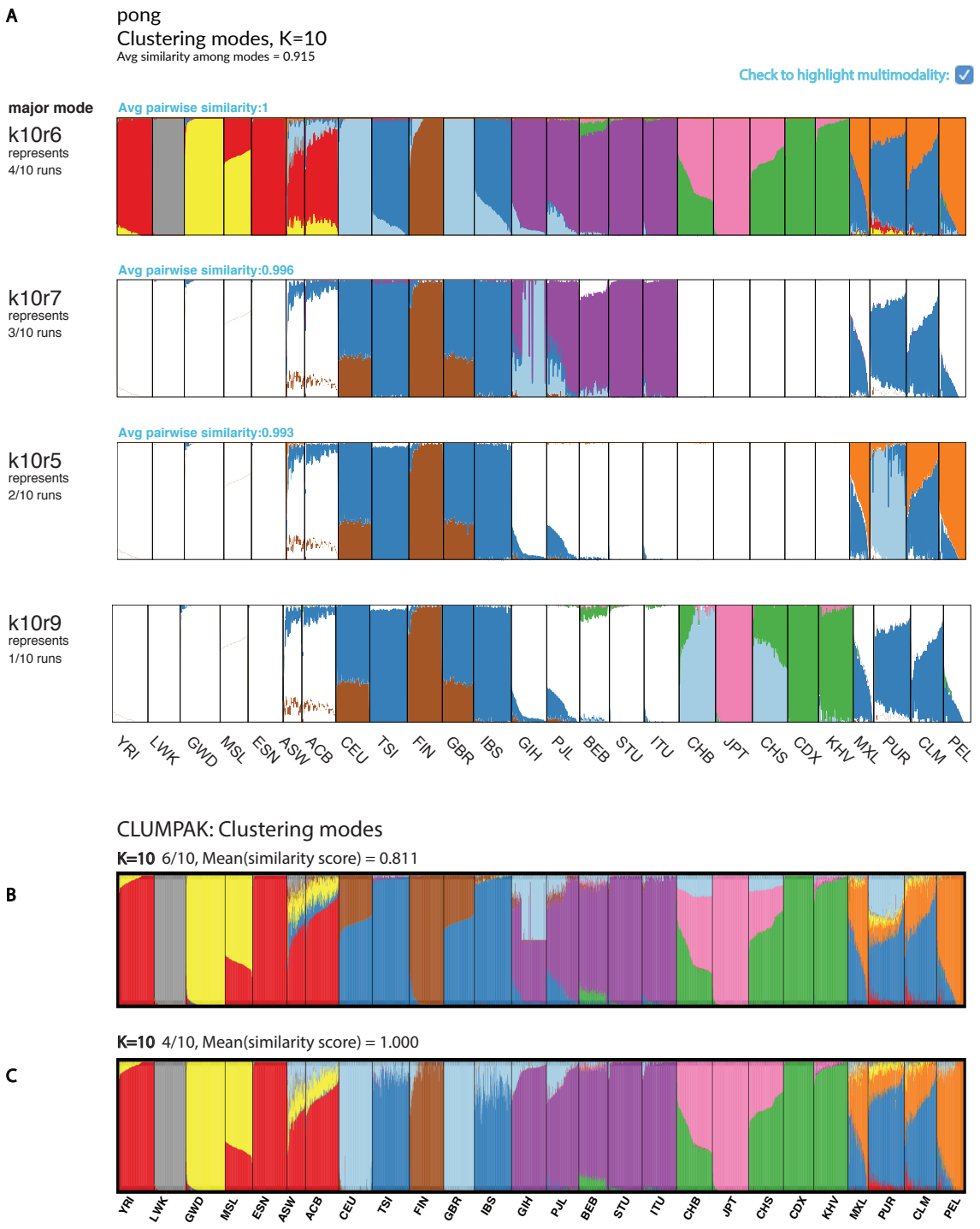


Figure 3: Visualizations of modes in population structure identified by pong and CLUMPAK at $K = 10$ for clustering inference with ADMIXTURE (Alexander et al., 2009) applied to 1000 Genomes data (phase 3; Consortium (2015)). The new cluster of membership coefficients first identified at $K = 10$ is denoted by light blue in each barplot. **A:** pong's dialog box of modes at $K = 10$, with multimodality highlighted. **B:** CLUMPAK's major mode at $K = 10$ averages over six runs of clustering inference output; the reported mean similarity score among these six runs is 0.811. South Asian (GIH), Han Chinese (CHB and CHS), and Puerto Rican (PUR) individuals all have ancestry depicted by the light blue cluster in this plot. The six runs averaged here are instead partitioned into three minor modes by pong in panel A. **C:** CLUMPAK's minor mode at $K = 10$ averages over four identical runs (mean similarity score is 1.000). This barplot contains the same information as the barplot of k4r10, representing pong's major mode in panel A.

these are distinct modes. We note that the highest-likelihood value of K for the 1000 Genomes data we analyzed is $K = 8$; at that value of K, we also see that CLUMPAK's major mode suggests shared ancestry among individuals that are actually identified as having non-overlapping membership coefficients when individual runs are examined (Supplementary Figures S1, S2).

5 Discussion

Here we introduce pong, a freely available user-friendly network-graphical method for post-processing output from clustering inference using population genetic data. We demonstrate that pong accurately aligns Q matrices orders of magnitude more quickly than do existing methods; it also provides a detailed characterization of modes among runs and produces a customizable, interactive D3.js visualization securely displayed using a localhost. pong's algorithm deviates from existing approaches by finding the maximum-weight perfect matching between column vectors of membership coefficients for pairs of Q matrices, and leveraging the Hungarian algorithm to solve this series of optimization problems (Kuhn, 1955, 1956; Munkres, 1957).

Interpreting the results from multiple runs of clustering inference is a difficult process. Investigators often choose a single Q matrix at each value of K to display or discuss, overlooking complex signals present in their data because the process of producing the necessary visualizations is too time-consuming. pong's speed allows the investigator to focus instead on conducting more runs of clustering inference in order to fully interpret the clustering in her sample of interest. Currently, many population-genetic studies only carry out one run of clustering inference per value of K, particularly when using ADMIXTURE's maximum-likelihood approach (Alexander et al., 2009) to the inferential framework implemented in STRUCTURE (Pritchard et al., 2000). The likelihood landscape of the input genotype data is complex, and thus ADMIXTURE can identify different local maxima across different runs for a given value of K (see Verdu et al. (2014)). Combining pong's rapid algorithm and detailed, interactive visualization with posterior probabilities for K reported by clustering inference methods will allow investigators to accurately interpret results from clustering inference, thereby advancing our knowledge of the genetic structure of natural populations of a wide range of organisms. We further plan to extend pong to visualize results from other applications of mixed-membership models and to leverage the dynamic nature of bound data to increase the information provided by pong's visualization.

Acknowledgements

Thanks to the Ramachandran Lab for helpful discussions. Data sets for testing early versions of pong were provided by Elizabeth Atkinson and Brenna Henn, Charleston Chiang and John Novembre, Caitlin Uren, and Paul Verdu; the Henn and Novembre labs, Chris Gignoux, and Catherine Luria tested beta versions of pong. We also thank Mark Howison for helpful discussions regarding python packaging. Multiple members of the Raphael Lab at Brown University helped improve pong, especially Max Leiserson and Hsin-Ta Wu, who advised on D3.js, and Mohammed El-Kebir, whose suggestions increased the efficiency of the back end and improved the manuscript.

Funding: This work was supported by a Brown University Undergraduate Teaching and Research Award (UTRA) to AAB, and a Research Experiences for Undergraduates Supplement to a National Science Foundation Faculty Early Career Development Award [DBI-1452622 to SR]. SR is a Pew Scholar in the Biomedical Sciences supported by The Pew Charitable Trusts, and an Alfred P. Sloan Research Fellow.

Figure S1: pong's main visualization of major modes in population structure in the 1000 Genomes (phase 3) with detailed population labels, $K = 2$ through $K = 10$.

Figure S2: CLUMPAK's (Kopelman et al., 2015) visualization of modes in population structure in the 1000 Genomes (phase 3), $K = 2$ through $K = 10$.

References

Alexander, D. H., Novembre, J., and Lange, K. (2009). Fast model-based estimation of ancestry in unrelated individuals. Genome Research, **19**(9), 1655–1664.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet Allocation. Journal of Machine Learning Research, **3**(4-5), 993–1022.

Bryc, K., Auton, A., Nelson, M. R., Oksenberg, J. R., Hauser, S. L., Williams, S., Froment, A., Bodo, J.-M., Wambebe, C., Tishkoff, S. a., and Bustamante, C. D. (2010). Genome-wide patterns of population structure and admixture in West Africans and African Americans. Proceedings of the National Academy of Sciences of the United States of America, **107**(2), 786–91.

Consortium, T. . G. P. (2015). A global reference for human genetic variation. Nature, **526**(7571), 68–74.

Falush, D., Stephens, M., and Pritchard, J. K. (2003). Inference of population structure using multilocus genotype data: linked loci and correlated allele frequencies. Genetics, **164**(4), 1567–87.

Galanter, J. M., Fernandez-Lopez, J. C., Gignoux, C. R., Barnholtz-Sloan, J., Fernandez-Rozadilla, C., Via, M., Hidalgo-Miranda, A., Contreras, A. V., Figueroa, L. U., Raska, P., Jimenez-Sanchez, G., Zolezzi, I. S., Torres, M., Ponte, C. R., Ruiz, Y., Salas, A., Nguyen, E., Eng, C., Borjas, L., Zabala, W., Barreto, G., González, F. R., Ibarra, A., Taboada, P., Porras, L., Moreno, F., Bigham, A., Gutierrez, G., Brutsaert, T., León-Velarde, F., Moore, L. G., Vargas, E., Cruz, M., Escobedo, J., Rodriguez-Santana, J., Rodriguez-Cintrón, W., Chapela, R., Ford, J. G., Bustamante, C., Seminara, D., Shriver, M., Ziv, E., Burchard, E. G., Haile, R., Parra, E., and Carracedo, A. (2012). Development of a panel of genome-wide ancestry informative markers to study admixture throughout the americas. PLoS Genetics, **8**(3), e1002554.

Glover, K. a., Quintela, M., Wennevik, V., Besnier, F., Sø rvik, A. G. E., and Skaala, O. y. (2012). Three decades of farmed escapees in the wild: A spatio-temporal analysis of atlantic salmon population genetic structure throughout norway. PLoS ONE, **7**(8), e43129.

Hubisz, M. J., Falush, D., Stephens, M., and Pritchard, J. K. (2009). Inferring weak population structure with the assistance of sample group information. Molecular Ecology Resources, **9**(5), 1322–1332.

Huelsenbeck, J. P., Andolfatto, P., and Huelsenbeck, E. T. (2011). Structurama: bayesian inference of population structure. Evolutionary Bioinformatics, **7**, 55–9.

Jakobsson, M. and Rosenberg, N. a. (2007). CLUMPP: a cluster matching and permutation program for dealing with label switching and multimodality in analysis of population structure. Bioinformatics (Oxford, England), **23**(14), 1801–6.

Jasra, A., Holmes, C., and Stephens, D. (2005). Markov Chain Monte Carlo methods and the label switching problem in Bayesian Mixture Modeling. Statistical Science, **20**(1), 50–67.

Kopelman, N. M., Mayzel, J., Jakobsson, M., Rosenberg, N. A., and Mayrose, I. (2015). C LUMPAK : a program for identifying clustering modes and packaging population structure inferences across K. Molecular Ecology Resources, pages doi: 10.1111/1755–0998.12387.

Kuhn, H. W. (1955). The Hungarian Method for the assignment problem. Naval Research Logistics Quarterly, **2**, 83–97.

Kuhn, H. W. (1956). Variants of the Hungarian method for assignment problems. Naval Research Logistics Quarterly, **3**, 253–258.

Manber, U. (1989). Introduction to Algorithms: A Creative Approach. Addison-Wesley.

Moore, A. J., Moore, W. L., and Baldwin, B. G. (2014). Genetic and ecotypic differentiation in a californian plant polyploid complex (Grindelia, Asteraceae). PLoS ONE, **9**(4), e95656.

Moreno-Estrada, A., Gignoux, C. R., Fernández-López, J. C., Zakharia, F., Sikora, M., Contreras, A. V., Acuña Alonzo, V., Sandoval, K., Eng, C., Romero-Hidalgo, S., Ortiz-Tello, P., Robles, V., Kenny, E. E., Nuño Arana, I., Barquera-Lozano, R., Macín-Pérez, G., Granados-Arriola, J., Huntsman, S., Galanter, J. M., Via, M., Ford, J. G., Chapela, R., Rodriguez-Cintron, W., Rodríguez-Santana, J. R., Romieu, I., Sienra-Monge, J. J., del Rio Navarro, B., London, S. J., Ruiz-Linares, A., Garcia-Herrera, R., Estrada, K., Hidalgo-Miranda, A., Jimenez-Sanchez, G., Carnevale, A., Soberón, X., Canizales-Quinteros, S., Rangel-Villalobos, H., Silva-Zolezzi, I., Burchard, E. G., and Bustamante, C. D. (2014). The genetics of Mexico recapitulates Native American substructure and affects biomedical traits. Science (New York, N.Y.), **344**(6189), 1280–5.

Munkres, J. (1957). Algorithms for the Assignment and Transportation Problems. Journal of the Society of Industrial and Applied Mathematics, **5**(1), 32–38.

Novembre, J. (2014). Variations on a common STRUCTURE: new algorithms for a valuable model. Genetics, **197**(3), 809–811.

Patterson, N., Price, A. L., and Reich, D. (2006). Population structure and eigenanalysis. PLoS Genetics, **2**(12), e190.

Price, A. L., Patterson, N. J., Plenge, R. M., Weinblatt, M. E., Shadick, N. a., and Reich, D. (2006). Principal components analysis corrects for stratification in genome-wide association studies. Nature Genetics, **38**(8), 904–909.

Pritchard, J. K., Stephens, M., and Donnelly, P. (2000). Inference of population structure using multilocus genotype data. Genetics, **155**(2), 945–959.

Raj, A., Stephens, M., and Pritchard, J. K. (2014). fastSTRUCTURE: Variational Inference of Population Structure in Large SNP Data Sets. Genetics, **197**(2), 573–589.

Rosenberg, N. a. (2004). DISTRUCT: A program for the graphical display of population structure. Molecular Ecology Notes, **4**(1), 137–138.

Stephens, M. (2000). Dealing with label switching in mixture models. J. R. Statist. Soc. Series B, **62**(4), 795–809.

Verdu, P., Pemberton, T. J., Laurent, R., Kemp, B. M., Gonzalez-Oliver, A., Gorodezky, C., Hughes, C. E., Shattuck, M. R., Petzelt, B., Mitchell, J., Harry, H., William, T., Worl, R., Cybulski, J. S., Rosenberg, N. a., and Malhi, R. S. (2014). Patterns of Admixture and Population Structure in Native Populations of Northwest North America. PLoS genetics, **10**(8), e1004530.

Supplementary Text for "pong: a network-graphical approach for the analysis of population structure"

Aaron A. Behr [1,2,*], Gracie Liu-Fang[3,†], Katherine Z. Liu[2,†], Priyanka Nakka[1,4] , and Sohini Ramachandran[1,4*]

[1]Department of Ecology and Evolutionary Biology, Brown University, Providence, RI, USA
[2]Department of Computer Science, Brown University, Providence, RI, USA
[3]Computer Science Department, Wellesley College, Wellesley, MA, USA
[4]Center for Computational Molecular Biology, Brown University, Providence, RI, USA

Processing of 1000 Genomes Data

Variant calls for 1,019,196 genome-wide single-nucleotide variants (SNVs) in 2,504 individuals were extracted from the 1000 Genomes Project Phase 3 data repository `ftp://ftp.1000genomes.ebi.ac.uk/vol1/ftp/release/20130502/` (release date: Nov 6, 2014), using the command-line tool `tabix` (Li, 2011).

A total of 78 individuals were excluded from analysis based on relatedness: one individual from each pair of first- and second-degree relatives was removed, leaving a total of 2,426 individuals. Next, SNVs were pruned for linkage disequilibrium using the `-indep-pairwise` flag in PLINK (Purcell et al., 2007). We removed every SNV with $r^2 > 0.1$ with any other SNV within a 50-SNV sliding window (PLINK command-line parameters for `-indep-pairwise`: 50 10 0.01), leaving a total of 225,705 SNVs for analysis.

ADMIXTURE (Alexander et al., 2009) was applied 10 times per value of K to these data, with K taking on values in the closed interval $[2, 10]$. The value of K that minimized cross-validation error was $K = 8$.

References

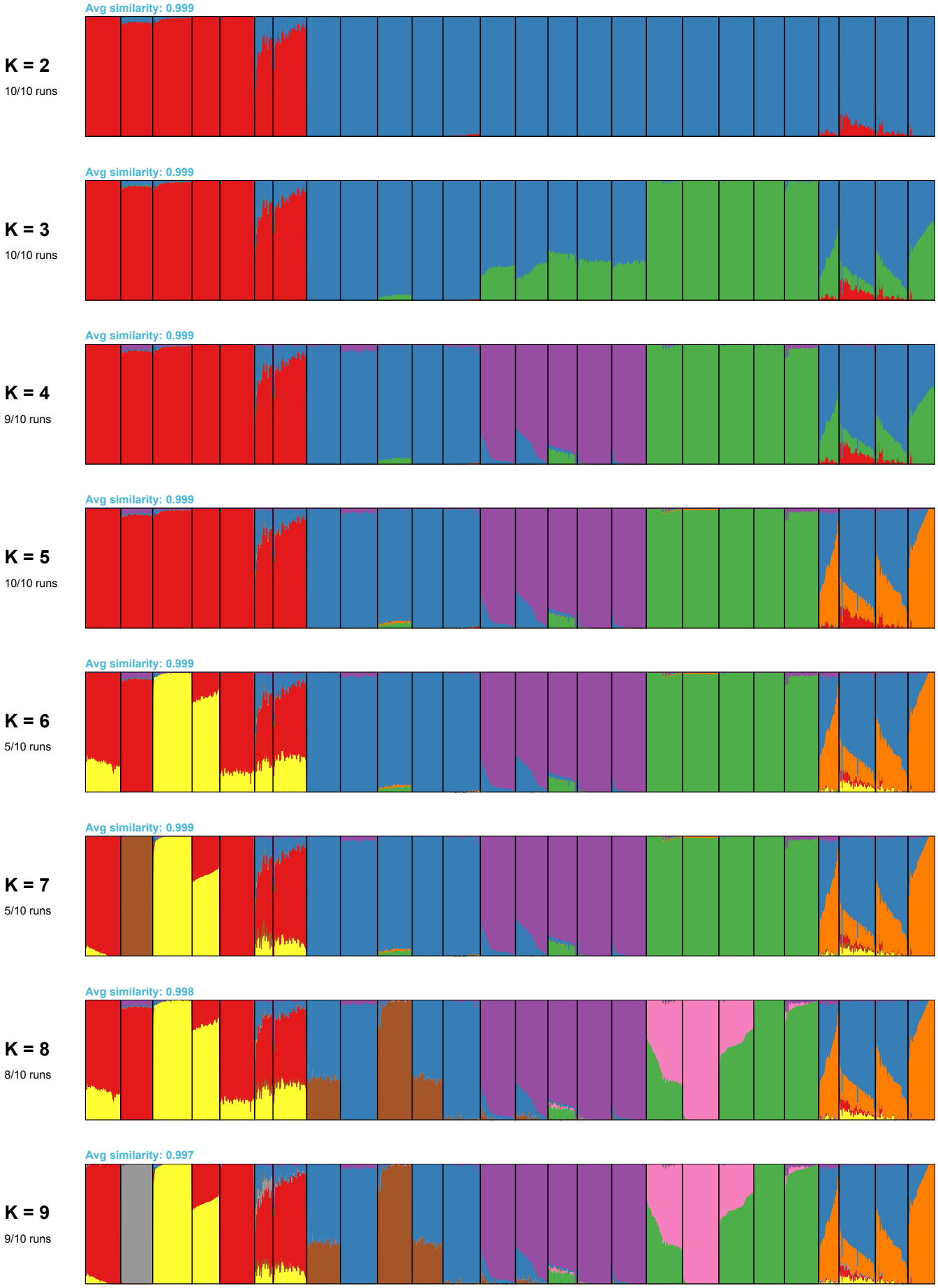
Alexander, D. H., Novembre, J., and Lange, K. (2009). Fast model-based estimation of ancestry in unrelated individuals. Genome Research, **19**(9), 1655–1664.

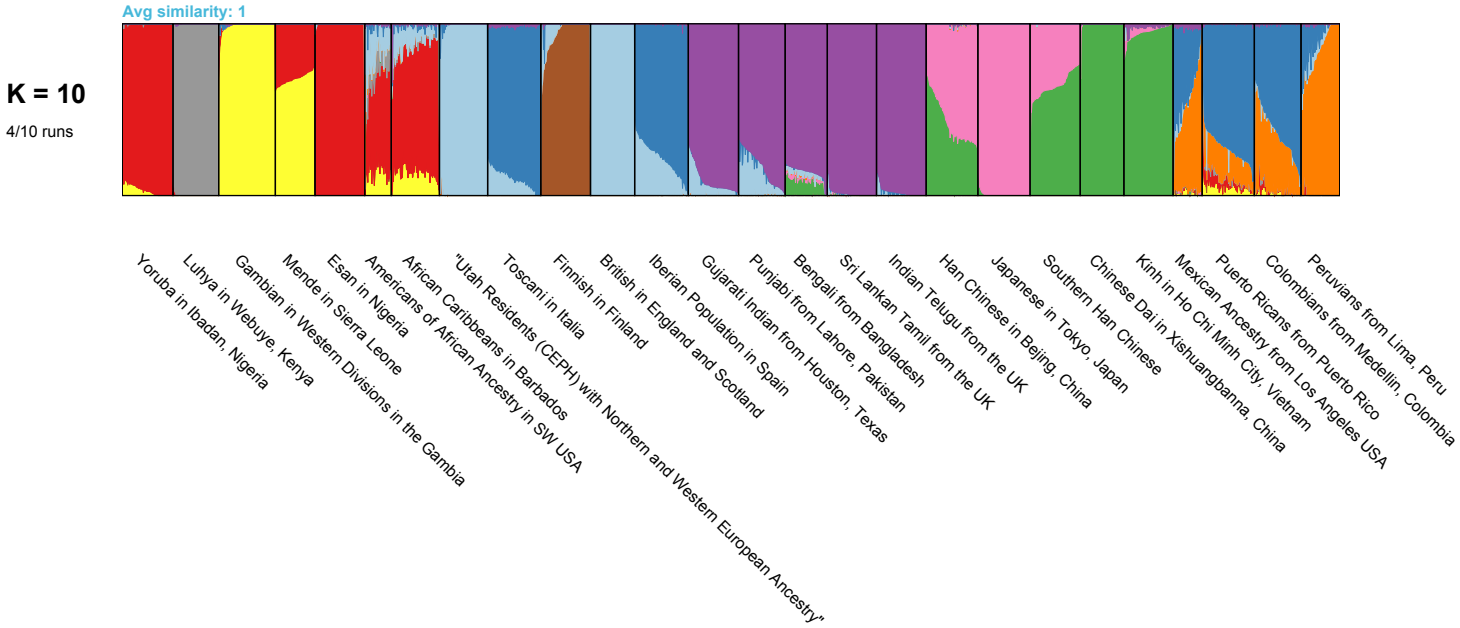
Li, H. (2011). Tabix: fast retrieval of sequence features from generic TAB-delimited files. Bioinformatics, **27**, 718–9.

Purcell, S., Neale, B., Todd-Brown, K., Thomas, L., Ferreira, M. A., Bender, D., Maller, J., Sklar, P., De Bakker, P. I., Daly, M. J., and Sham, P. C. (2007). PLINK: a tool set for whole-genome association and population-based linkage analyses. The American Journal of Human Genetics, **81**(3), 559–575.

*To whom correspondence should be addressed: aaron_behr@alumni.brown.edu, sramachandran@brown.edu
†These authors contributed equally to this work.

Figure S1

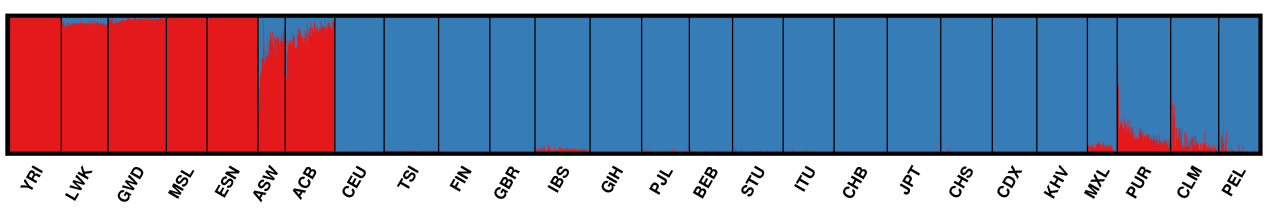

Avg similarity: 0.999
K = 2
10/10 runs
Avg similarity: 0.999
K = 3
10/10 runs
Avg similarity: 0.999
K = 4
9/10 runs
Avg similarity: 0.999
K = 5
10/10 runs
Avg similarity: 0.999
K = 6
5/10 runs
Avg similarity: 0.999
K = 7
5/10 runs
Avg similarity: 0.998
K = 8
8/10 runs
Avg similarity: 0.997
K = 9
9/10 runs


Avg similarity: 1
K = 10
4/10 runs
Yoruba in Ibadan, Nigeria
Luhya in Webuye, Kenya
Gambian in Western Divisions in the Gambia
Mende in Sierra Leone
Esan in Nigeria
Americans of African Ancestry in SW USA
African Caribbeans in Barbados
"Utah Residents (CEPH) with Northern and Western European Ancestry"
Toscani in Italia
Finnish in Finland
British in England and Scotland
Iberian Population in Spain
Gujarati Indian from Houston, Texas
Punjabi from Lahore, Pakistan
Bengali from Bangladesh
Sri Lankan Tamil from the UK
Indian Telugu from the UK
Han Chinese in Bejing, China
Japanese in Tokyo, Japan
Southern Han Chinese
Chinese Dai in Xishuangbanna, China
Kinh in Ho Chi Minh City, Vietnam
Mexican Ancestry from Los Angeles USA
Puerto Ricans from Puerto Rico
Colombians from Medellin, Colombia
Peruvians from Lima, Peru

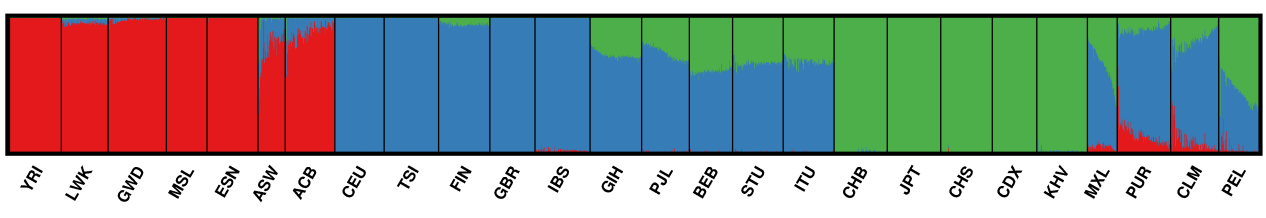
CLUMPAK main pipeline - Job 1439007297 summary

Major modes for the uploaded data:

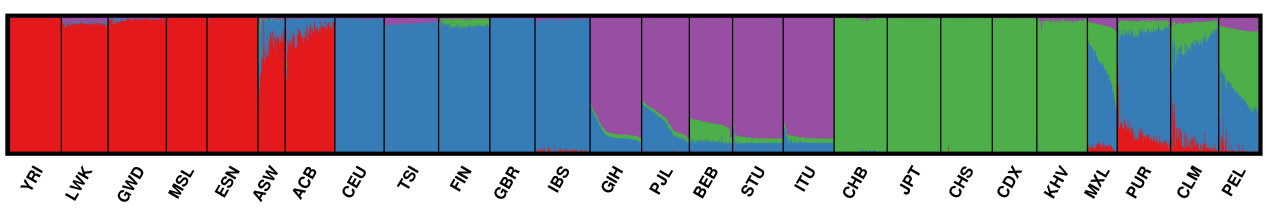
K=2



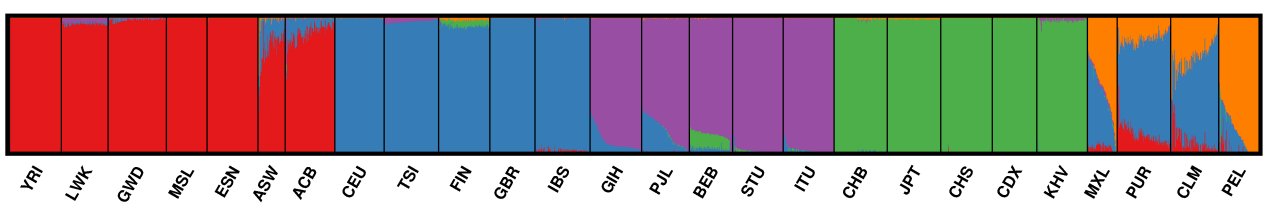
K=3



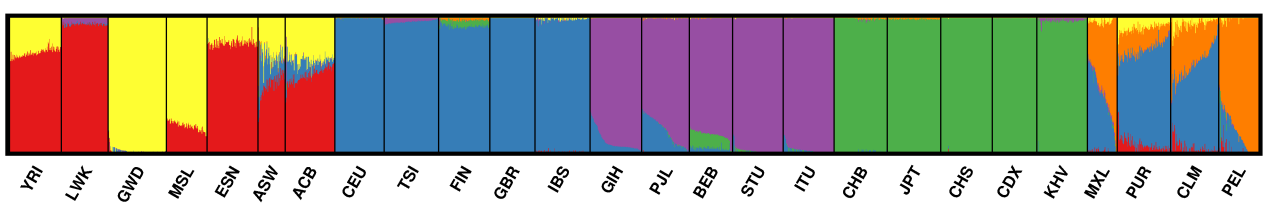
K=4



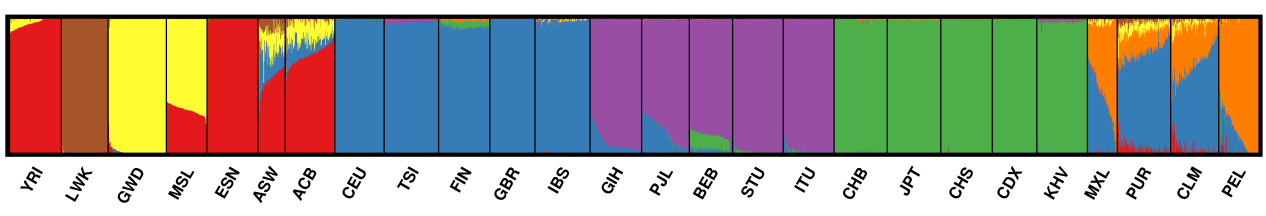
K=5



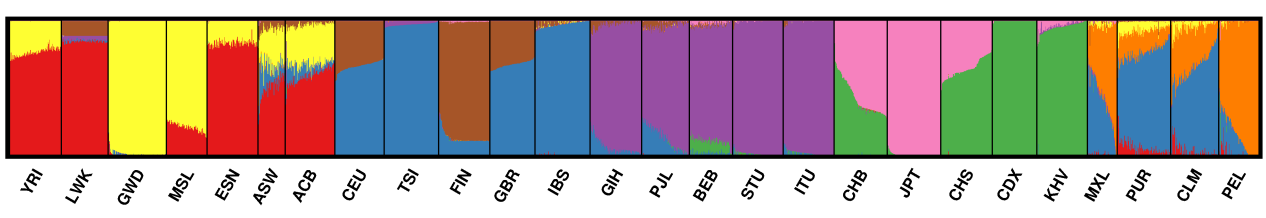
K=6



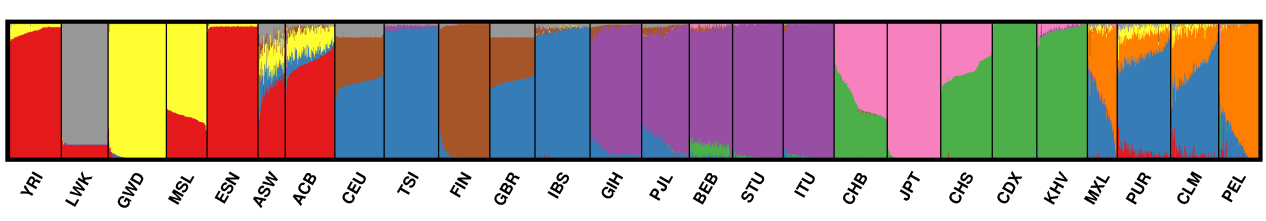
K=7



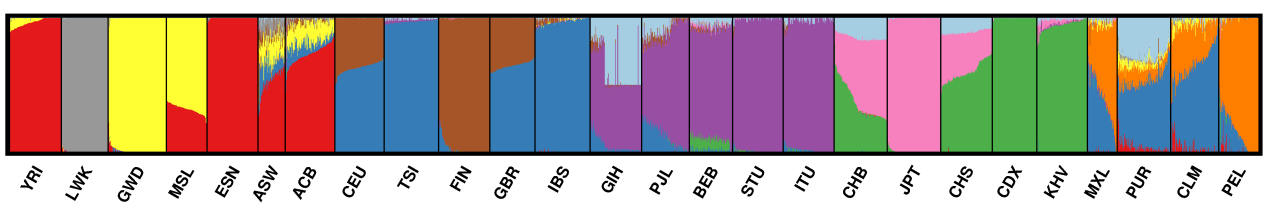
K=8



K=9

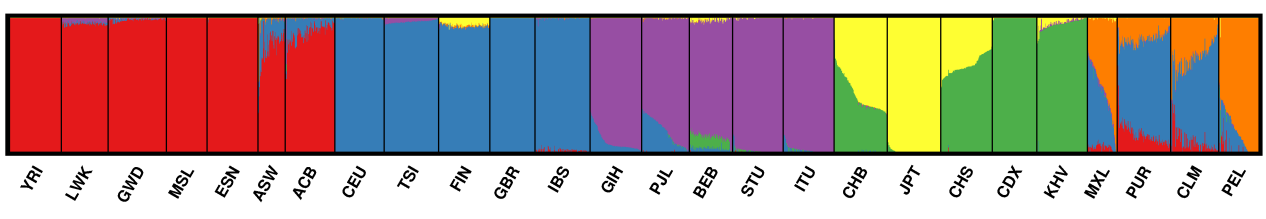


K=10

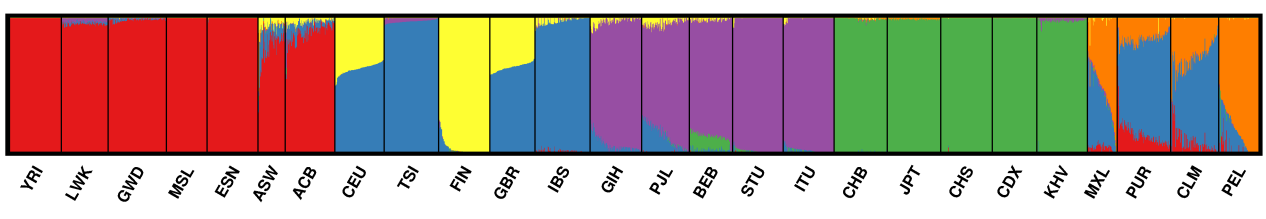


Minor modes for the uploaded data:

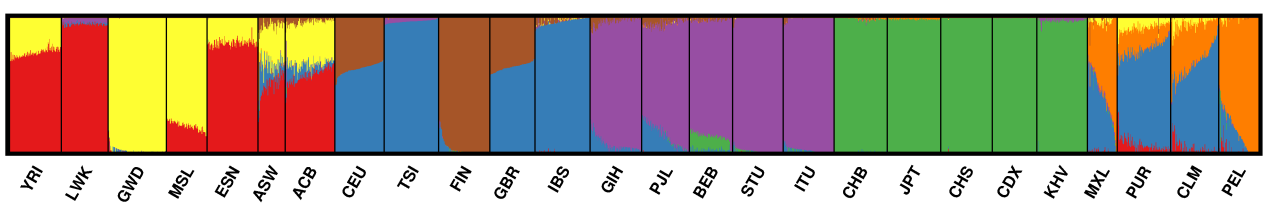
K=6 MinorCluster1



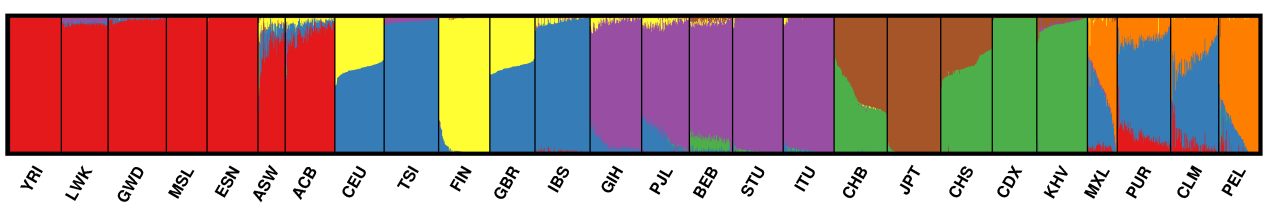
K=6 MinorCluster2



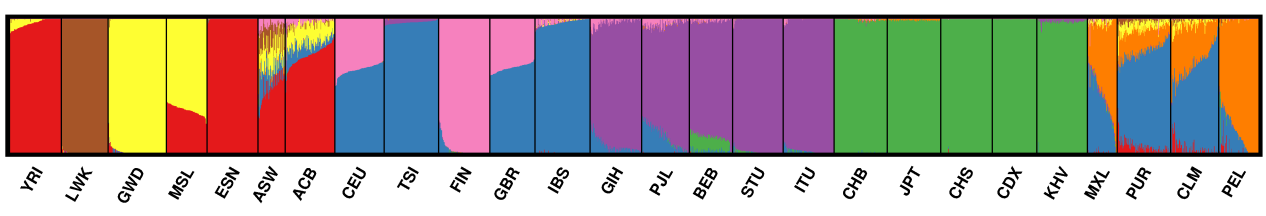
K=7 MinorCluster1



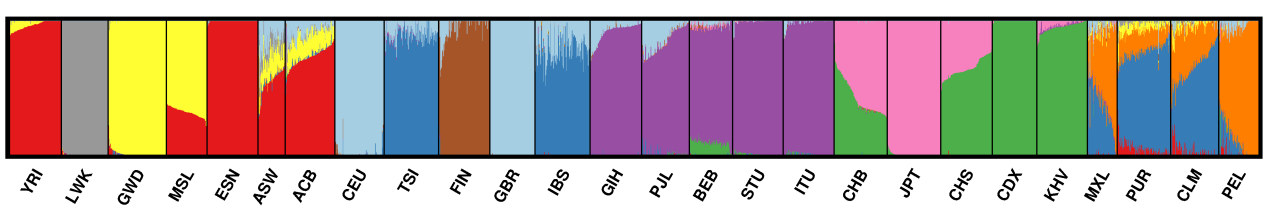
K=7 MinorCluster2



K=8 MinorCluster1



K=10 MinorCluster1



Division of runs by mode:

K=2	10/10
K=3	10/10
K=4	10/10
K=5	10/10
K=6	5/10, 3/10, 2/10
K=7	5/10, 3/10, 2/10
K=8	9/10, 1/10
K=9	10/10
K=10	6/10, 4/10